compacting said array in a manner that for any consecutive element S_j^k and S_p^q, removing S_j^k from said array if the memory address of said start point S_j^k is equal to the memory address of said start point S_p^q, and removing said start point S_p^q from said array if the next hop of said start point S_j^k is equal to said next hop of said start point S_p^q;

A4

removing each element S_j^k from said array when said memory address of said start point S_j^k is equal to the memory address of the end point E_0^0; and

concurrently constructing a Compression Bit Map Array and a Compressed Next Hop Array by assigning "1" for each start point in said array to generate said Compression Bit Map Array and assigning an next hop of said start point in said array to each corresponding entry of the Compressed Next Hop Array to generate said Comrpessed Next Hop Array.

A5

20. (Amended) The method as claimed in claim 17, wherein each of said Next Hop Array contains 2^k entries, where k is determined by the longest prefix length of each segment.

REMARKS

The present amendment is in response to the Office Action mailed January 30, 2001, in which Claims 1, 3, 5, and 7 through 19 were allowed and Claims 2, 4, 6 and 20 were rejected. Applicants have thoroughly reviewed the outstanding Office Action including the Examiner's remarks and the reference

cited therein. The following remarks are believed to be fully responsive to the Office Action and, when coupled with the amendments made herein, are believed to render all claims at issue patentably distinguishable over the cited references.

Claims 2, 4, 6, 8 and 20 are amended herein. No claims are cancelled. No claims are added. Accordingly, Claims 1 through 20 remain pending.

All the changes are made for clarification and are based on the application and drawings as originally filed. It is respectfully submitted that no new matter is added.

Applicants respectfully request reconsideration in light of the above amendments and the following remarks.

CLAIM REJECTIONS – 35 U.S.C. SECTION 112

With respect to Paragraphs 3 and 4 of the Office Action, the Examiner rejected Claims 2, 4, 6 and 20 under 35 U.S.C. Section 112, 2^{nd} paragraph as being infinite for failing to particularly point out and distinctly claim the subject matter of the present invention.

Applicants respectfully traverse these rejections.

Applicants have amended these claims for clarification respectively. More specifically, proper antecedent bases in Claims 2, 4 and 6 have been made. The spelling errors of "assigning" are corrected in Claim 8. Claim 20 is

amended as being dependent to claim 17 as well.

Having overcome the rejections in the Office Acton, withdrawal of the rejections and expedited passage of the application to issue are respectfully requested.

MARKED-UP CHANGES

Attached hereto is a marked-up version of the changes made to the claims by the current amendment. The attached paper is captioned "**VERSION WITH MARKINGS TO SHOW CHANGES MADE.**"

CONCLUSION

In light of the above amendments and remarks, Applicants respectfully submit that all pending Claims 1 through 20 as currently presented are in condition for allowance. If, for any reason, the Examiner disagrees, please call the undersigned attorney at 202-624-3947 in an effort to resolve any matter still outstanding *before* issuing another action. The undersigned attorney is confident that any issue which might remain can readily be worked out by telephone.

Favorable reconsideration is respectfully requested.

Respectfully submitted,

Thomas T. Moga
Registration No. 34,881
Attorney for Applicants

POWELL, GOLDSTEIN, FRAZER & MURPHY, LLP
P.O. Box 97233
Washington, D.C. 20090-7223
202-624-3947

RECEIVED
MAY 0 1 2002
OFFICE OF PETITIONS

Dated: **April 22, 2002**

TTM/hs

VERSION WITH MARKINGS TO SHOW CHANGES MADE

(USSN 09/240,833)

IN THE CLAIMS:

Claims 2, 4, 6, 8 and 20 have been amended as follows:

2. (Amended) The method as claimed in claim 1, further comprising the steps of:

using a first bit stream of an IP address of an incoming IP packet as an index to look up said segmentation table;

determining if said first bit stream of an entry of said segmentation table pointed to by said index is larger than or equal to M;

outputting said entry of said segmentation table as a next hop when said first bit stream of said entry is smaller than said M.

determining [said] a correspondent entry of said segmentation table is a pointer pointing to a corresponding Next Hop Array when said first bit stream of said entry of said segmentation table is larger than or equal to said M, and [said] a second bit stream of said corresponding entry of said segmentation table is smaller than or equal to Y, and using said second bit stream of said corresponding entry of said segmentation table as an index to look up said corresponding Next Hop Array.

4. (Amended) The method as claimed in claim 1, further comprising the steps of:

determining [said] a correspondent entry of said segmentation table is a pointer pointing to a corresponding Code Word Array when [said] a first bit stream of said entry is larger than or equal to [said] M, and [said] a second bit stream of said correspondent entry of said segmentation table is larger than said Y;

computing an index for looking up a corresponding code word in said Code Word Array by adding said correspondent entry of said segmentation table, which is a pointer, plus said second bit stream;

computing an index for looking up a corresponding Compressed Next Hop Array by adding (said pointer + $2^{k-4} \times 4 - 1$), a Base of said corresponding Code Word and $|w|$, said $|w|$ representing the number of "1"s accumulated from the *0-th* bit to *w-th* bit of the map of the code word of said corresponding Code Word Array, and said *k* representing an offset length.

6. (Amended) The method as claimed in claim 1, wherein each of said Next Hop [Array] Arrays contains 2^k entries, and said k is determined by the longest prefix length of each segment.

8. (Amended) The method as claimed in claim 7 comprises the steps of:

reading a set of sorted route prefixes of a segment in an increasing order by the length of prefixes, and each pair of start point and end point of said list of sorted route prefixes is sorted according to the order in said set of route prefixes of said segment ;

sorting each element in said set in an order according to its memory

address in said segment;

processing each element in said set from left to right and in a manner that:

(a) determining if a selected element is a start point S_i^0, where "i" represents the *i-th* route prefix in said set and "0" represents the number of update times of said start point in the memory; when said selected element is a start point S_i^0, executing step (b), and when said selected element is not a start point S_i^0, executing step (c);

(b) pushing said start point S_i^0 onto a stack, and appending said start point S_i^0 to an array; repeate said step (a) until each element in said set is processed;

(c) removing a top element of said stack;

(d) determining if the top element of said stack is a start point S_j^k, where "*j*" represents the *j-th* route prefix in said set and "k" represents the number of update times of said start point in the memory; when said top element is said start point S_j^k, executing step (e), and when said top element is not said start point S_j^k, executing step (f);

(e) appending S_j^{k+1} to said array where the next hop of a start point S_j^{k+1} is equal to the next hop of a start point S_j^k, and the memory address of a start point S_j^{k+1} is equal to the memory address of an end address E_i^0+1; and replacing the top element of said stack with said start point S_j^{k+1}; repeate said step (a) until each element in said

set is processed;

(f) executing nothing; repeate said setp (a) until each element in said set is processed;

compacting said array in a manner that for any consecutive element S_j^k and S_p^q, removing S_j^k from said array if the memory address of said start point S_j^k is equal to the memory address of said start point S_p^q, and removing said start point S_p^q from said array if the next hop of said start point S_j^k is equal to said next hop of said start point S_p^q;

removing each element S_j^k from said array when said memory address of said start point S_j^k is equal to the memory address of the end point E_0^0; and

concurrently constructing a Compression Bit Map Array and a Compressed Next Hop Array by [assinging] assigning "1" for each start point in said array to generate said Compression Bit Map Array and [assinging] assigning an next hop of said start point in said array to each corresponding entry of the Compressed Next Hop Array to generate said Comrpessed Next Hop Array.

20. (Amended) The method as claimed in claim 17, wherein each of said Next Hop [Array] Arrays contains 2^k entries, where k is determined by the longest prefix length of each segment.